Mining Local Patterns from Fuzzy Temporal Data

Fokrul Alom Mazarbhuiya

Abstract— **Mining patterns from datasets having fuzzy time attributes is an important data mining problem. Some of these mining task are finding locally frequent sets, local association rules etc Most of the earlier works were mainly devoted on mining non-fuzzy temporal datasets .In this article, we propose a method extracting locally frequent itemsets from fuzzy temporal datasets. The efficacy of the method is established with the help of an experiment conducted on a synthetic dataset.**

***Index Terms*— Core of a fuzzy number, Data mining, frequent sets, fuzzy membership function, α-cut.**

I. INTRODUCTION

Mining association rules from datasets has been defined initially [1] by R. Agarwal *et al* for application in large super markets datasets. Super market datasets are temporal in the sense that every transaction in dataset is associated with the time of transaction. Mining frequent itemsets from such dataset is an important data mining problem.

In this paper, we consider datasets, where the time of transaction is imprecise i.e. fuzzy temporal. Such datasets are termed as fuzzy temporal. In [2], authors proposed a method of finding locally frequent itemsets from such dataset. The work done in [2] is mainly theoretical and little work have been done on the implementation side. In this article, we propose to work on the implementation of the algorithm [2]. The implementation is a trie-based implementation. We use here a synthetic dataset to show the efficacy of the algorithm.

The paper is organized as follows: In section-II we give a brief discussion on the recent works in Temporal Data Mining and fuzzy temporal data mining. In section-III we describe the terms and notations used in this paper. In section-IV, we give the algorithm proposed [2]. In section-V, we give the implementation detail along with experimental results. We conclude with conclusion and lines for future work in section-VI.

II RECENT WORKS

Agrawal et al [1] formulated the problem of association rules discovery. In[1], a method for the discovery of association rules was given, which is known as the A priori algorithm. Temporal Data Mining is now an important extension of conventional data mining and has recently been able to attract more people to work in this area. Considering the time aspect, more interesting patterns time dependent can be extracted. Primarily there are two broad directions of temporal data mining [3]. One concerns the extraction of causal relationships among temporally oriented events. The other concerns the extraction of similar patterns within the same time sequence or among different time sequences. The name sequence mining is normally used for the later problem. In [4] the authors discussed the problem of recognizing frequent episodes in an event sequence.

The association rule discovery process has been extended by incorporating temporal aspects. In temporal association rules each rule has associated with it a time interval in which the rule holds. The problem is to find frequent itemsets which are frequent certain valid time periods then extracting rules from such frequent itemsets. In [5], [6], [7] and [8], the problem of temporal data mining is addressed and techniques and algorithms have been developed for this. In [9] an algorithm for the discovery of temporal association rules is described.. In [10, 11], an efficient method for finding locally and periodically frequent sets and periodic association rules are discussed. In [2], authors proposed a method of finding locally frequent itemsets from fuzzy temporal data.

III TERMS,NOTATION AND SYMBOL USED

A Some Definitions related to Fuzziness

A fuzzy interval is actually a fuzzy number with a flat region. We denote a fuzzy interval A is by $A = [a, b, c, d]$ where $a < b < c < d$ and $A(a) = A(d) = 0$, $A(x) = 1$ for all $x \in [b, c]$. $A(x)$ for all $x \in [a, b]$ is known as left reference function and $A(x)$ for $x \in [c, d]$ is known as the right reference function. [12].

An α-cut of the fuzzy interval $[t_1\text{-}a, t_1, t_2, t_2\text{+}a]$ is a closed interval $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$.

The core of a fuzzy number A is the set of elements of A having membership value one i.e.

$\text{Core}(A) = \{(x, A(x); A(x) = 1\}$

For every fuzzy set A,

$$A = \bigcup_{\alpha \in [0,1]} {}_{\alpha}A$$

where ${}_{\alpha}A(x) = \alpha.\ {}^{\alpha}A(x)$, and ${}_{\alpha}A$ is a special fuzzy set.

For any two fuzzy sets A and B and for all $\alpha \in [0, 1]$,

$${}^{\alpha}(A \cup B) = {}^{\alpha}A \cup {}^{\alpha}B$$

$${}^{\alpha}(A \cap B) = {}^{\alpha}A \cap {}^{\alpha}B$$

For any two fuzzy numbers A and B, the membership functions $A(x)$ and $B(x)$ are said to be similar similar to each other if the slope of the left reference function of $A(x)$ is equal to the that of $B(x)$ and the slope of right reference of $A(x)$ is equal that of $B(x)$. Thus for any two fuzzy numbers A and B having similar membership functions

$$\left| {}^{\alpha}A \right| = \left| {}^{\alpha}B \right|, \forall \alpha \in [0, 1]$$

B Some Definitions related to Association Rule Mining over Fuzzy time period

Suppose that $T = <t_o, t_1, \ldots\ldots\ldots>$ is a sequence of imprecise or fuzzy time stamps over which a linear ordering $<$ is defined. We also assume that all the fuzzy time stamps are having similar membership functions. Let I be a finite set of items and the transaction dataset D is a collection of transactions with the property that each transaction has two

Manuscript received.
Fokrul Alom Mazarbhuiya, College of Computer Science and IT, Albaha University, Albaha, KSA

parts, one subset of the itemset I and the other fuzzy time-stamp indicating the approximate time in which the transaction had taken place. We assume that D is ordered in the ascending order of the core of fuzzy time stamps. A transaction is said to be in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ if the α-cut of the fuzzy time stamp of the transaction is contained in α-cut of $[t_1\text{-}a, t_1, t_2, t_2+a]$ for some user's specified value of α.

The local support of an itemset in a fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ is defined as the ratio of the number of transactions in the time interval $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$ containing the itemset to the total number of transactions in $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$ for the whole dataset D for a given value of α. The notation $Sup_{[t_1-a,t_1,t_2,t_2+a]}(X)$ is used to denote the support of the itemset X in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$. Given a threshold σ we say that an itemset X is frequent in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ if $Sup_{[t_1-a,t_1,t_2,t_2+a]}(X) \geq (\sigma/100)$* tc where tc denotes the total number of transactions in D that are in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$.

IV ALGORITHM PROPOSED

A Generating Locally Frequent Sets

Before proceeding further, for the sake of convenience, we describe the algorithm proposed in [2].

While constructing locally frequent sets, with each locally frequent set a list of fuzzy time-intervals is maintained in which the set is frequent. Two user's specified thresholds α and *minthd* are used for this. During the execution of the algorithm while making a pass through the database, if for a particular itemset the α-cut of its current fuzzy time-stamp, $[^{\alpha}Lcurrent, {}^{\alpha}Rcurrent]$ and the α-cut, $[^{\alpha}Llastseen, {}^{\alpha}Rlastseen]$ of its fuzzy time, when it was last seen overlap then the current transaction is included in the current time-interval under consideration which is extended with replacement of $^{\alpha}Rlastseen$ by $^{\alpha}Rcurrent$; otherwise a new time-interval is started with $^{\alpha}Lcurrent$ as the starting point. The support count of the item set in the previous time interval is checked to see whether it is frequent in that interval or not and if it is so then it is fuzzified and added to the list maintained for that set. Also for the locally frequent sets over fuzzy time intervals, a minimum core length of the fuzzy period is given by the user as *minthd* and fuzzy time intervals of core length greater than or equal to this value are only kept. If *minthd* is not used than an item appearing once in the whole database will also become locally frequent a over fuzzy point of time.

Procedure to compute L_1, the set of all locally frequent item sets of size 1.

For each item while going through the database we always keeps an α-cut $^{\alpha}lastseen$ which is $[^{\alpha}Llastseen, {}^{\alpha}Rlastseen]$ that corresponds to the fuzzy time stamp when the item was last seen. When an item is found in a transaction and the fuzzy time-stamp is *tm* and if its α-cut $^{\alpha}tm=[^{\alpha}Ltm, {}^{\alpha}Rtm]$ has empty intersection with $[^{\alpha}Llastseen, {}^{\alpha}Rlastseen]$, then a new time interval is started by setting *start* of the new time interval as $^{\alpha}Ltm$ and *end* of the previous time interval as $^{\alpha}Rlastseen$. The previous time interval is fuzzified provided the support of the item is greater than *min-sup*. The fuzzified interval is then added to the list maintained for that item provided that the duration of the core is greater than *minthd*. Otherwise $^{\alpha}Rlastseen$ is set to $^{\alpha}Rtm$, the counters maintained for counting transactions are increased appropriately and the process is continued.

Following is the algorithm to compute L_1, the list of locally frequent sets of size 1. Suppose the number of items in the dataset under consideration is n and we assume an ordering among the items.

1)

2) Algorithm 1

$C_1 = \{(i_k, tp[k]) : k = 1,2,.....,n\}$

where i_k is the k-th item and tp[k] points to a list of fuzzy time intervals initially empty.}

for k = 1 to n do

set $^{\alpha}lastseen[k] = \phi$;

set itemcount[k]and transcount[k] to zero for each transaction t in the database with fuzzy time stamp tm

do

{for k = 1 to n do

{ if $\{i_k\} \subseteq t$ then

{ if($^{\alpha}lastseen[k] == \phi$)

{$^{\alpha}lastseen[k] = {}^{\alpha}firstseen[k] = {}^{\alpha}tm$;

itemcount[k] = transcount[k] = 1;

}

else

if($[^{\alpha}Llastseen[k], {}^{\alpha}Rlastseen[k]] \cap [^{\alpha}Ltm[k], {}^{\alpha}Rtm[k]] \neq \phi$)

{$^{\alpha}Rlastseen[k] = {}^{\alpha}Rtm[k]$; itemcount[k]++;

transcount[k]++;

}

else

*{ if (itemcount[k]/transcount[k]*100 $\geq \sigma$)*

fuzzify($[^{\alpha}Llastseen[k], {}^{\alpha}Rlastseen[k]], \forall \alpha \in [0, 1]$)

if(|core(fuzzified interval)|$\geq$ minthd)

add(fuzzified interval) to tp[k];

itemcount[k] = transcount[k] = 1;

lastseen[k] = firstseen[k] = tm;

}

}

else transcount[k]++;

} // end of k-loop //

} // end of do loop //

for k = 1 to n do

*{ if (itemcount[k]/transcount[k]*100$\geq\sigma$)*

fuzzify($[^{\alpha}Llastseen[k], {}^{\alpha}Rlastseen[k]], \forall \alpha \in [0, 1]$)

if(|core(fuzzified interval)|$\geq$ minthd)

add(fuzzified interval) to tp[k];

if(tp[k] != 0) add $\{i_k, tp[k]\}$ to L_1

}

fuzzify($[^{\alpha}a, {}^{\alpha}b], \alpha$)

{ fuzzified interval= $\bigcup_{\alpha \in [0,1]} {}_{\alpha}[a, b]$;

where ${}_{\alpha}[a, b](x) = \alpha.\ {}^{\alpha}[a, b](x)$

return(fuzzified interval)

}

Two support counts are kept, *itemcount* and *transcount*. If the count percentage of an item in an α-cut of a fuzzy time interval is greater than the minimum threshold then only the set is considered as a locally frequent set over fuzzy time interval.

L_1 as computed above will contain all 1-sized locally frequent sets over fuzzy time intervals and with each set there is associated an ordered list of fuzzy time intervals in which

the set is frequent. Then A priori candidate generation algorithm is used to find candidate frequent set of size 2. With each candidate frequent set of size two we associate a list of fuzzy time intervals that are obtained in the pruning phase. In the generation phase this list is empty. If all subsets of a candidate set are found in the previous level then this set is constructed. The process is that when the first subset appearing in the previous level is found then that list is taken as the list of fuzzy time intervals associated with the set. When subsequent subsets are found then the list is reconstructed by taking all possible pair wise intersection of subsets one from each list. Sets for which this list is empty are further pruned.

Using this concept we describe below the modified A-priori algorithm for the problem under consideration.

Algorithm 2
Modified A priori

B. Initialize

k = 1;
C_1 = all item sets of size 1
L_1 = {frequent item sets of size 1 where
with each itemset {i_k} a list tp[k] is maintained which gives all time fuzzy
intervals in which the set is frequent}
*L_1 is computed using algorithm 1.1 */*
for(k = 2; $L_{k-1} \neq \phi$; k++) do
{ C_k = apriorigen(L_{k-1})
/ same as the candidate generation method of the A priori algorithm setting tp[i] to zero for all i*/*
prune(C_k);
drop all lists of fuzzy time intervals maintained with the sets in C_k
Compute L_k from C_k.
//L_k can be computed from C_k using the same procedure used for computing L_1 //
k = k + 1
}

Answer = $\bigcup_k L_k$

Prune(C_k)
{Let m be the number of sets in C_k and let the sets be $s_1, s_2, \ldots, s_m$. Initialize the pointers tp[i] pointing to the list of fuzzy time-intervals maintained with each set s_i to null
for i = 1 to m do
{for each (k-1) subset d of s_i do
{if d $\notin L_{k-1}$ then
{$C_k = C_k$ - {s_i, tp[i]}; break;}
else
{ if (tp[i] == null) then set tp[i] to point to the list of fuzzy time intervals maintained for d
else
{ take all possible pair-wise intersection of fuzzy time intervals one from each list,one list maintained with tp[i] and the other maintained with d and take this as the list for tp[i]
delete all fuzzy time intervals whose core length is less than the value of minthd if tp[i] is empty then {$C_k = C_k$ - {s_i,tp[i]};
break;
}
}
}
}
}

V IMPLEMENTATION

A Data structure used

Candidate generation, pruning and support count need an efficient data structures in which all candidates are stored. In general two data structures are used for this purpose namely hash-tree and trie data structure. In our work we used hash-tree data structure.

1 Hash tree data structures

The nicety about hash tree based implementation is that it reduces the number comparisons by storing the candidates in a hash tree. So, it makes the execution faster.

B Analysis of Results

For experimented conducted in the paper, we used a synthetic dataset available at http://fimi.cs.helsinki.fi/testdata.html. As the dataset does not have fuzzy time contents, we incorporate the same. We consider the different sizes of transactions like 10,000, 20,000, 30,000, 40,000, 50,000, 100,000 execute the algorithm. We keep the life time of dataset as one year i.e. year 2012. The results obtained by method given in table1 and figue1.

Transaction sizes	Number frequent itemsets
10,000	1
20,000	1
30,000	2
40,000	3
50,000	4
100,000	9

Table1: frequent itemsets extracted by the method [2]

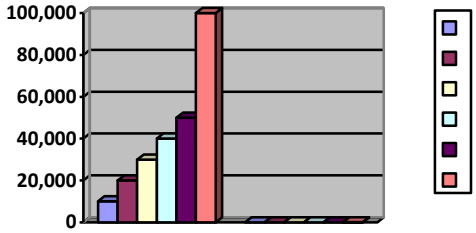

Figure1: frequent itemsets extracted by the method [2]

VI CONCLUSION

The algorithm [2] for finding frequent sets that are frequent in certain fuzzy time periods from fuzzy temporal data, is given in this paper. The algorithm dynamically extracts the frequent sets along with their fuzzy time intervals where the sets are frequent. These frequent sets are named as locally frequent sets over fuzzy time interval. The technique used is similar to the A priori algorithm. The efficacy of the algorithm is established with the help of experiment conducted on a synthetic data. The implementation is hash tree based implementation. In future, we will go for other type of implementation of the same algorithm.

References

[1] R. Agrawal, T. Imielinski and A. Swami; Mining association rules between sets of items in large databases; Proceedings of the ACM SIGMOD '93, Washington, USA (May 1993).

[2] F. A Mazharbhuiya, M. Shenify and Mohammed Husamuddin, Finding Local and Periodic Association Rules from Fuzzy Temporal Data, *The 2014 International Conference on Advances in Big Data Analytics* July 21-24, 2014, Las Vegas, Nevada, USA.

[3] J. F. Roddick, M. Spillopoulou; A Biblography of Temporal, Spatial and Spatio-Temporal Data Mining Research; ACM SIGKDD (June 1999).

[4] H. Manilla, H. Toivonen and I. Verkamo; Discovering frequent episodes in sequences; KDD'95; AAAI, 210-215 (August 1995).

[5] J. M. Ale and G.H. Rossi; An approach to discovering temporal association rules; Proceedings of the 2000 ACM symposium on Applied Computing (March 2000).

[6] X. Chen and I. Petrounias; A framework for Temporal Data Mining; Proceedings of the 9th International Conference on Databases and Expert Systems Applications, DEXA '98, Vienna, Austria. Springer-Verlag, Berlin; Lecture Notes in Computer Science 1460 (1998), 796-805.

[7] X. Chen and I. Petrounias; Language support for Temporal Data Mining; Proceedings of 2nd European Symposium on Principles of Data Mining and Knowledge Discovery, PKDD '98, Springer Verlag, Berlin (1998), 282-290.

[8] X. Chen, I. Petrounias and H. Healthfield; Discovering temporal Association rules in temporal databases; Proceedings of IADT'98 (International Workshop on Issues and Applications of Database Technology (1998), 312-319.

[9] J. M. Ale, and G. H. Rossi; An Approach to Discovering Temporal Association Rules, *In Proc. of 2000 ACM symposium on Applied Computing* (2000).

[10] A. K. Mahanta, F. A. Mazarbhuiya and H. K. Baruah; Finding Locally and Periodically Frequent Sets and Periodic Association Rules, Proceeding of 1st Int'l Conf on Pattern Recognition and Machine Intelligence (PreMI'05),LNCS 3776 (2005), 576-582.

[11] F. A. Mazarbhuiya Yusuf Pervaiz (2015); An Efficient Method for Generating Local Association Rules, International Journal of Applied Information Systems (IJAIS), Foundation of Computer Science FCS, New York, USA Volume 9 – No.2, June 2015.

[12] D. Dubois and H. Prade; Ranking fuzzy numbers in the setting of possibility theory, *Information Science* 30(1983), 183-224.

Fokrul Alom Mazarbhuiya, College of Computer Science and IT, Albaha University, Albaha, KSA